Chapter 3

Dynamic Programming Alignment Accuracy

3.1 Introduction

Alignments of biological sequences generated by computational algorithms are routinely used as a basis for inference about sequences whose structure or function is unknown. The standard approach is to find the best-scoring alignment between a pair of sequences, where the the score rewards aligning similar residues, and penalises substitutions and gaps. The best-scoring alignment can be found by *dynamic programming* [NW70]. Other approaches that are frequently used, such as FASTA [LAK89] and BLAST [AG96], approximate this.

An important question for a biologist faced with the results of such a program is: How accurate is the proposed alignment? It is clearly desirable that an alignment algorithm return the most accurate alignment it can, but the notion of alignment accuracy implies the existence of a "correct" alignment, the definition of which is non-trivial. One approach is to construct a definitive structural alignment (based on crystallographic data and/or human judgement) which can then be compared with alignments returned by the algorithms in question. However, this is a difficult process to automate and it is not always clear what is really wanted biologically.

Another approach is to take a closer look at the inherent properties of the alignment algorithm itself. One can view the algorithm as a system for identifying the relationships between two sequences which have diverged due to random mutations (substitutions and indels) [TKF92]. By repeatedly simulating the experiment of randomly mutating a pair of initially identical sequences, then feeding the two sequences into the alignment algorithm, one can obtain a measure of the accuracy of the algorithm. In this paper the results of such empirical experiments are fist given. A theoretical estimate of the accuracy is then developed, and shown to provide a good approximation to the observed behaviour. A table from which accuracy values can be predicted for commonly used scoring systems is also given. Finally it is described how to calculate the expected accuracy of a given alignment, and how this can be used to construct

an optimal accuracy alignment algorithm which performs demonstrably better than standard dynamic programming.

Other attempts to quantify and predict the accuracy of alignments have mainly been empirical and have focused on multiple alignments [MVF94], [Got96]. Mevissen and Vingron [MV96] have addressed pairwise alignment reliability recently, and Hwa, Lässig and Drasdo have developed theoretical approaches complementing those presented here [HL96, DHL97b].

3.2 Definitions and notation

This chapter will consider in detail the global alignment in which the entire length of the two input sequences must be aligned [NW70], although most of the results obtained will be equally applicable to the corresponding algorithms for local alignment [SW81].

3.2.1 Definition of the alignment fidelity

In this chapter, a pairwise alignment $\mathbf{a}$ between two sequences ($\mathbf{X}$,$\mathbf{Y}$) is described by the set of aligned residues or *couplings* ($i \diamond j$) between residue i of $\mathbf{X}$ and residue j of $\mathbf{Y}$.

Given a correct alignment $\mathbf{a}_{\text{real}}$, define the fidelity $F(\mathbf{a})$ of $\mathbf{a}$ as the fractional overlap between $\mathbf{a}$ and $\mathbf{a}_{\text{real}}$, i.e.:

$$F(\mathbf{a}) = \frac{|\mathbf{a} \cap \mathbf{a}_{\text{real}}|}{|\mathbf{a}_{\text{real}}|} \tag{3.1}$$

This corresponds to the partial overlap fraction metric defined in Chapter 2.

3.2.2 Choice of scoring parameters

Let us first treat the simplest biologically-relevant case: global alignment of two DNA sequences ($\mathbf{X}$,$\mathbf{Y}$) with linear gap costs and a "flat" substitution matrix (one that doesn't differentiate between e.g. purine-purine and purine-pyrimidine substitutions). The score $S_{\mathbf{a}}$ for a particular alignment $\mathbf{a}$ is then:

$$S_{\mathbf{a}} = a\alpha + b\beta + c\gamma$$

where a, b and c are (respectively) the number of match, mismatch and gap columns in the alignment $\mathbf{a}$, and α, β and γ are match, mismatch and gap scores (typically but not necessarily with $\alpha > 0$, $\beta < 0, \gamma < 0$).

Although the score $S_{\mathbf{a}}$ depends on three free parameters (α, β and γ), the maximum scoring alignment $\mathbf{a}_{\mathbf{max}}$ only depends on one effective parameter. To see this, note first that global alignments must account for every residue in $\mathbf{X}$ and $\mathbf{Y}$, and so:

$$2a + 2b + c = L_{\mathbf{X}} + L_{\mathbf{Y}}$$

where $L_{\mathbf{X}}$ and $L_{\mathbf{Y}}$ are the lengths of $\mathbf{X}$ and $\mathbf{Y}$. Now consider the transformed score $S'_{\mathbf{a}}$:

$$S'_{\mathbf{a}} = \frac{S_{\mathbf{a}} - \frac{\alpha}{2}(L_{\mathbf{X}} + L_{\mathbf{Y}})}{\alpha - \beta} = -b - c\lambda$$

where

$$\lambda = \frac{\alpha/2 - \gamma}{\alpha - \beta} \tag{3.2}$$

Since $S'_{\mathbf{a}}$ differs from $S_{\mathbf{a}}$ only by an offset and a scaling factor, both of which are independent of the particular alignment $\mathbf{a}$, it follows that the ordering of the scores $S_{\mathbf{a}}$ of all possible alignments $\mathbf{a}$ (and hence the choice of maximally-scoring alignment $\mathbf{a}_{\mathbf{max}}$) is determined uniquely by λ.

The parameter λ can be considered to be an effective gap penalty. When $\lambda \gg \frac{1}{2}$, then $\beta \gg 2\gamma$ and the highest-scoring alignment will be minimally gapped as mismatches will be favoured over gaps. When $0 < \lambda < \frac{1}{2}$, then $\beta < 2\gamma < \alpha$ and gap regions will score higher than mismatches, with the consequence that all substitutions will be misidentified as pairs of indels. When $\lambda \leq 0$, then $2\gamma \geq \alpha$ (assuming $\alpha > \beta$) and gap regions will score higher than matches, which is clearly disastrous [VW94].

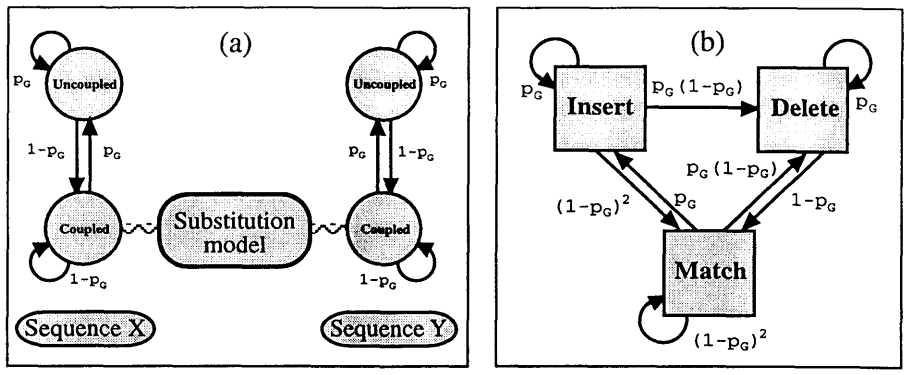

Figure 3.1: (a) Coupled Markov model of sequence evolution. Each sequence is represented by a semi-independent Markov chain, coupled by a point substitution model. (b) The corresponding finite state automaton for sequence alignment.

3.2.3 Probabilistic interpretation

Figure 3.1a shows a probabilistic model of sequence evolution that will be seen to correspond to the alignment algorithm described in Section 3.2.2. Each sequence is modelled by a hidden Markov chain with two states, labelled *coupled* and *uncoupled*. When both sides of the model are in the coupled state, aligned residues are emitted in pairs, one on each side. When either side is in the uncoupled state, unaligned residues are emitted singly on that side. Coupled emissions stem from a common ancestral residue; the joint probability distribution for the residue pair is derived from a point substitution model. Uncoupled emissions are unaligned and independent. Transitions from the coupled into the uncoupled state occur with probability p_G, as do self-looping transitions in the uncoupled state. (N.B. for affine gaps, the *coupled→uncoupled* transition still has probability p_G, but the self-looping *uncoupled→uncoupled* transition is assigned the independent gap-extension probability p_E.) The independence of the two Markov chains is restricted by the requirement that neither chain is allowed to enter the coupled state on its own (both must enter it simultaneously).

3.2.4 A simple point substitution model

For the experiments described below, the following simplified one-parameter model of nucleotide substitution was used. Start with identical residue pairs, one in each sequence, chosen at random from the set {A,C,T,G}. For each of the two residues, replace it with a randomly-chosen nucleotide with probability p_S. The replacement nucleotide has a one in four chance of being identical to the residue it is replacing. The probability q_{XY} of the residue pair (X, Y) being emitted in the coupled state is thus:

$$q_{XY} = \begin{cases} \frac{1}{16}(1 + 3(1 - p_S)^2) & \text{if } X = Y \\ \frac{1}{16}(1 - (1 - p_S)^2) & \text{if } X \neq Y \end{cases} \tag{3.3}$$

The probability q_X of the residue X being emitted in the uncoupled state is:

$$q_X = \frac{1}{4} \tag{3.4}$$

Note that if

$$p_S = 1 - e^{-2kt}$$

where k is a point substitution rate and t is a time-like parameter, this model is identical to that proposed by Jukes and Cantor [JC69].

3.2.5 Relationship between probabilistic model and alignment algorithm

Figure 3.1b depicts a stochastic finite-state machine for traversing the combined state space of the coupled Markov chains of Figure 3.1a. The *match* state of the automaton in Figure 3.1b emits coupled residue pairs in both sequences, whereas the *insert* and *delete* states emit uncoupled residues in **X** and **Y** respectively. Note the asymmetry of the *insert*→*delete* transition, which is required to preserve the independence of the gap length distributions in each sequence.

The automaton in Figure 3.1b is itself a hidden Markov model, albeit one which models two sequences rather than one. Alignment of sequences to hidden Markov models is performed using the Viterbi dynamic programming algorithm. To identify the most likely alignment **a** for a pair of sequences related under the simple indel model, one uses the Viterbi algorithm to align the sequences to the automaton in Figure 3.1b. This turns out to be mathematically equivalent to the standard (Needleman-Wunsch) alignment algorithm; that is, Needleman-Wunsch finds the most likely set of ancestral residue couplings under the probabilistic mutation model given a pair of sequences (**X**,**Y**).

Assuming the substitution model described in Section 3.2.4, and using the scoring notation of Section 3.2.2, it is found that the alignment score $S_{\mathbf{a}}$ is equal to the posterior log-likelihood of the sequence pair if the following match, mismatch and gap scores are chosen:

$$\alpha = \log \frac{(1-p_G)^2(1+3(1-p_S)^2)}{16} \tag{3.5}$$

$$\beta = \log \frac{(1-p_G)^2(1-(1-p_S)^2)}{16} \tag{3.6}$$

$$\gamma = \log \frac{p_G}{4} \tag{3.7}$$

If one is not interested in the exact score of the alignment obtained, but only in ensuring that its score is maximised, and if one restricts oneself to global alignments, then one need only specify a single scoring parameter such as the parameter λ defined in (3.2). Denote by $\hat{\lambda}$ the probabilistic value for λ, which is obtained by substituting equations (3.5)-(3.7) into equation (3.2):

$$\hat{\lambda} = \frac{\log\left[(\frac{1}{p_G}-1)\sqrt{1+3(1-p_S)^2}\right]}{\log\left[\frac{1+3(1-p_S)^2}{1-(1-p_S)^2}\right]} \tag{3.8}$$

Given that $\hat{\lambda}$ returns the alignment with the highest log-likelihood under the generative model, it is natural to predict that it is the optimal value of λ for

reconstructing the correct alignment, in the sense that it maximises the fidelity $F(\mathbf{a}_{\max})$.

3.3 Results

3.3.1 Simulation 1: Optimisation of the alignment fidelity with respect to the scoring scheme

In order to test the prediction that $\hat{\lambda}$ is optimal, 50 pairwise alignments were randomly generated, each with 1000 aligned residue pairs plus gap regions, according to the evolutionary model of Section 3.2.3 with p_G and p_S set to a range of different values. The pairs of sequences thus generated were then independently re-aligned by the Needleman-Wunsch algorithm using a range of different values of λ, and the fidelities of the returned alignments were measured. With this procedure the value of λ that is optimal for reconstructing the alignment can be estimated and compared with the value $\hat{\lambda}$ predicted by equation (3.8).

3.3.2 Simulation 2: Measurement of the alignment fidelity

The sequence generation procedure of simulation 1 was performed at various different values of p_G and p_S and the sequences re-aligned using $\lambda = \hat{\lambda}$. The fidelity was measured and the process repeated until the mean re-alignment fidelity was known to within an error margin of ± 0.1 (this was a 95% confidence limit, assuming the fidelity of an alignment to be a Gaussian distributed random variable).

3.3.3 The probabilistic prediction $\hat{\lambda}$ is supported experimentally

Figure 3.2 shows values of $\hat{\lambda}$ for different values of p_G and p_S. Note that when $\hat{\lambda}$ drops below zero, effective reconstruction of the alignment is impossible, as gaps score higher than matches. This regime is indicated by the shaded region in Figure 3.2.

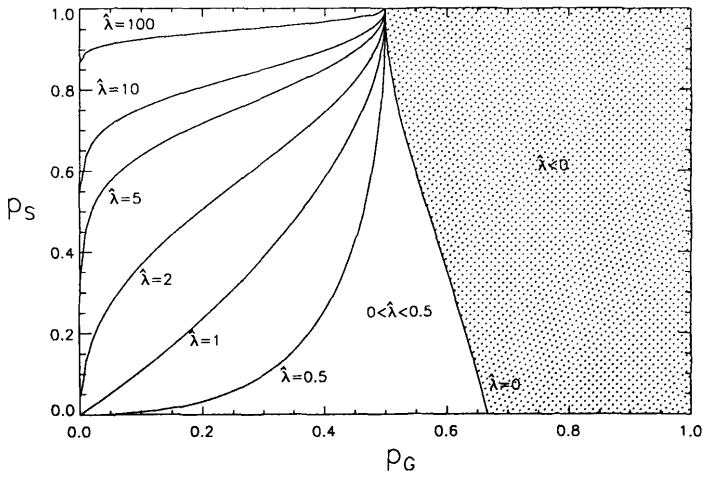

Figure 3.2: Contours of constant $\hat{\lambda}$ in mutation parameter space. $\hat{\lambda}$ is the effective gap penalty. The shaded region on the right-hand side of the plot represents $\hat{\lambda} < 0$, where pairs of indel events are more likely than matches and accurate alignment is effectively impossible.

Figure 3.3 shows how the fidelity F changes as a function of λ when $p_G = 0.1$ and $p_S = 0.2$. For $\lambda \leq 0$ the optimal alignment is all gaps and the fidelity is zero; for high λ the optimal alignment is minimally gapped and the fidelity flattens out, eventually reaching a plateau. In between these extremes there is a value of λ which maximises the fidelity.

By definition, setting $\lambda = \hat{\lambda}$ will find the most likely alignment, but there is no proof that this alignment will be the most faithful one. Figure 3.4 plots the observed optimal values of λ against the predicted values $\hat{\lambda}$. There is a good correspondence, supporting the hypothesis that the likelihood scoring approach is valid.

3.3.4 The fidelity decreases as p_G and p_S are increased

The graphs in Figure 3.5 show the dependence of the maximal fidelity F on the gap probability p_G and the substitution probability p_S. Figure 3.5a plots F as

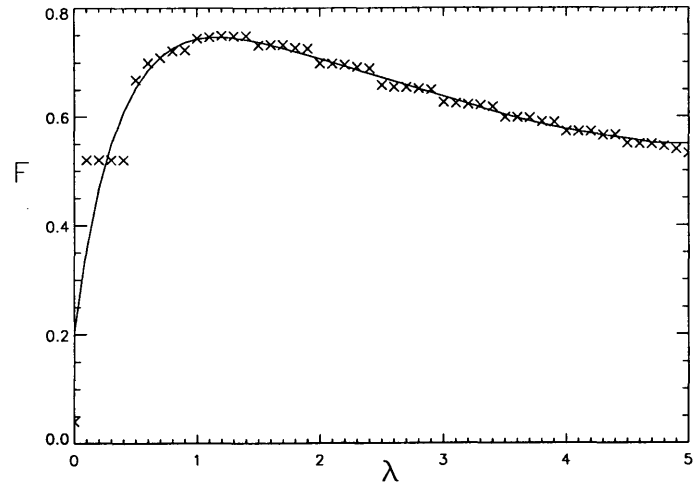

Figure 3.3: The fidelity F of alignments returned by dynamic programming for a range of values of the effective gap penalty λ, with p_G and p_S set to 0.1 and 0.2 respectively. When $\lambda \sim 0$, the optimal alignments are all gaps and $F \to 0$. As $\lambda \to \infty$, the optimal alignment tends to become minimally gapped, causing F to plateau. The data in this Figure are from simulation 1.

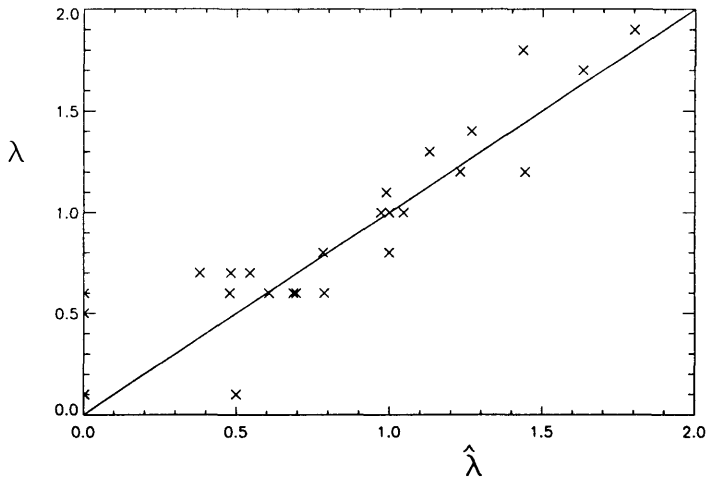

Figure 3.4: Values of λ which are observed from the simulation data to be optimal are compared with the values $\hat{\lambda}$ predicted by the likelihood scoring approach. There appears to be a strong correlation, with slope unity (solid line). The data in this Figure are from simulation 1.

a function of p_G at various different constant values of p_S and Figure 3.5b plots F against p_S at different constant values of p_G.

It can be seen that in general F decreases monotonically as the mutation parameters increase. The dependence of F on p_G and p_S is nearly linear up to around $(p_G, p_S) \sim (0.2, 0.2)$. Notable deviations from this behaviour are observable, for example at $(p_G, p_S) \simeq (0.2, 0.04)$ and again at $(p_G, p_S) \simeq (0.3, 0.1)$. At both these points the fidelity appears to be discontinuous. Referring back to Figure 3.2, it is seen that these points are on the locus $\lambda = 0.5$, which is recalled from Section 3.2.2 as the point at which mismatches become more likely than gaps. So the discontinuity can be identified with the scoring scheme entering a region of parameter space where substitution events are recognised.

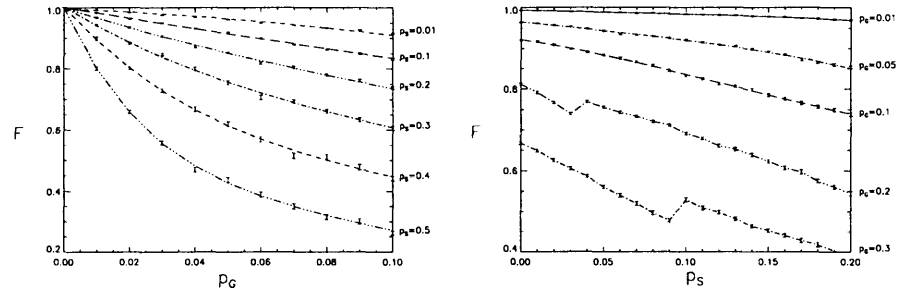

Figure 3.5: These graphs show the variation of the fidelity F (a) as a function of p_G at fixed p_S, and (b) as a function of p_S at fixed p_G. Note the discontinuities at $(p_G, p_S) \sim (0.2, 0.04)$ and $(0.3, 0.1)$, explained in the text. The data in this Figure are from simulation 2.

3.3.5 An analytic approximation to the alignment fidelity

Motivated by the near-linearity of the fidelity at low (p_G, p_S), an analytic approximation to the alignment fidelity can be developed.

To follow the analysis of the following section it is useful to be able to view an alignment geometrically, as a path through a dynamic programming matrix. The horizontal and vertical axes of the matrix represent the two aligned sequences $\mathbf{X}$ and $\mathbf{Y}$. A global alignment $\mathbf{a}$ is represented by a path from the top left to the bottom right of the matrix connecting all the coupled residue pairs $(x, y) \in \mathbf{a}$. Diagonal segments of the path correspond to match and mismatch regions and horizontal and vertical segments correspond to gaps. The fidelity of an alignment path $\mathbf{a}$ is its fractional overlap with the correct alignment path $\mathbf{a}_{\mathrm{real}}$.

When the mutation probabilities are small, the Viterbi alignment path $\mathbf{a}_{\mathrm{max}}$ returned by the dynamic programming algorithm is tightly bound to the correct path $\mathbf{a}_{\mathrm{real}}$. The main source of errors is misplacement of gaps by the algorithm, as illustrated in Figure 3.6. This effect is called *edge wander*. The fidelity in this regime is governed by the average displacement distance of each gap (the mean edge wander) and by the frequency of gaps. The next section describes

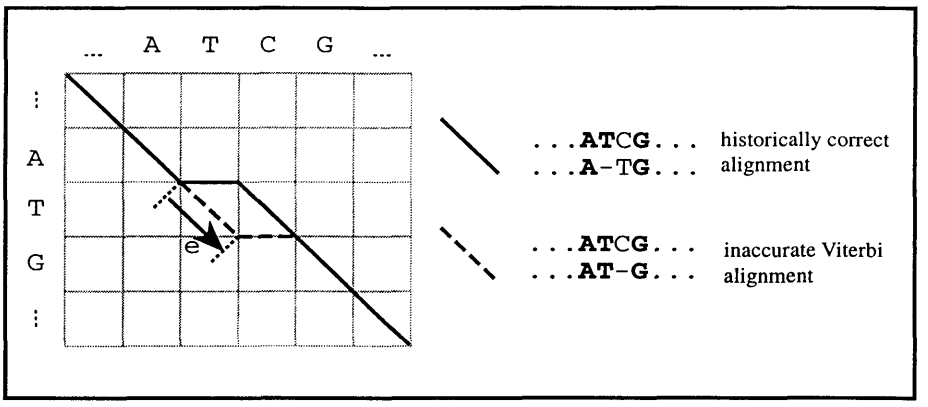

Figure 3.6: Edge wander - minor deviation of the Viterbi alignment path from the correct path - is the principal source of error in alignments between closely related sequences. In this toy example, the historically correct alignment (solid line) contains a mismatch next to an indel, but the Viterbi algorithm inevitably misaligns the two T residues (dotted line). The Viterbi edge wander e is defined to be the number of residues by which the gap is misplaced (here $e = 1$).

how to calculate the mean edge wander.

3.3.6 Calculation of the edge wander

Let the *edge wander* e be the displacement, in residues, of a gap in some near-perfect alignment **a** compared with the same gap in the correct alignment. Let $S(e)$ be the score of that segment of **a** which extends E residues to the left and right of the correct location of the gap, where E is some integer such that $(1/p_G) \gg E \gg e$. If v_k and w_k are the individual scores of the k'th residue pairings along adjacent diagonals (**v** and **w**) of the dynamic programming matrix, with $k = 0$ at the correct location of the gap (so that, in the notation of Section 3.2.1, v_k corresponds to residue pairing $(i + k \diamond j + k)$ and w_k to residue pairing $(i + k + 1 \diamond j + k)$, where i and j are such that the correct gap location sits between residue pairings $(i \diamond j)$ and $(i + 2 \diamond j + 1)$), then one can write:

$$
\begin{aligned}
S(e) &= \sum_{k=-E+1}^{e} v_k + \gamma + \sum_{k=e+1}^{E} w_k \\
&= \sum_{k=-E+1}^{E} w_k + \gamma + \sum_{k=-E+1}^{e} (v_k - w_k) \\
&= S_W + \gamma + \mathcal{R}(e)
\end{aligned}
$$

where γ is the gap score, $S_W = \sum_{k=-E+1}^{E} w_k$ is the score along diagonal **w**, and $\mathcal{R}(e) = \sum_{k=-E+1}^{e}(v_k - w_k)$ is the difference in score between alignment **a** and diagonal **w**, minus the gap penalty γ.

Note that since the v_k and w_k are independent random variables, $\mathcal{R}(e)$ is a Markov process. (Strictly, the series $(v_k, v_{k+1}, ...)$ is not independent of the series $(w_k, w_{k+1}, ...)$, since v_k and w_k represent residue pairings in the same row of the dynamic programming matrix. However, $\mathcal{R}(e)$ is still Markov.)

The v_k and w_k are not identically distributed for all k, since the correct path crosses over from **v** to **w** between $k = 0$ and $k = 1$. For convenience rewrite v_k and w_k in terms of the scores t_k and s_k of residue pairings on and off the correct path, respectively:

$$
v_k = \begin{cases} t_k & \text{if } k \leq 0 \\ s_k & \text{if } k > 0 \end{cases}
$$

$$
w_k = \begin{cases} s_k & \text{if } k \leq 0 \\ t_k & \text{if } k > 0 \end{cases}
$$

An expression for $\mathcal{R}(e)$ can now be written in terms of $r_k \equiv s_k - t_k$:

$$
\mathcal{R}(e) = \begin{cases} \sum_{k=-E+1}^{e}(-r_k) & \text{if } e \leq 0 \\ \sum_{k=-E+1}^{0}(-r_k) + \sum_{k=1}^{e} r_k & \text{if } e > 0 \end{cases} \tag{3.9}
$$

The random behaviour of $\mathcal{R}(e)$ is illustrated in Figure 3.7. On average, $\mathcal{R}(e)$ will be zero at $e = 0$ and negative elsewhere; in any specific case, however, the maximum of $\mathcal{R}(e)$ may be some distance away from $e = 0$ and this is where the alignment algorithm will place the gap.

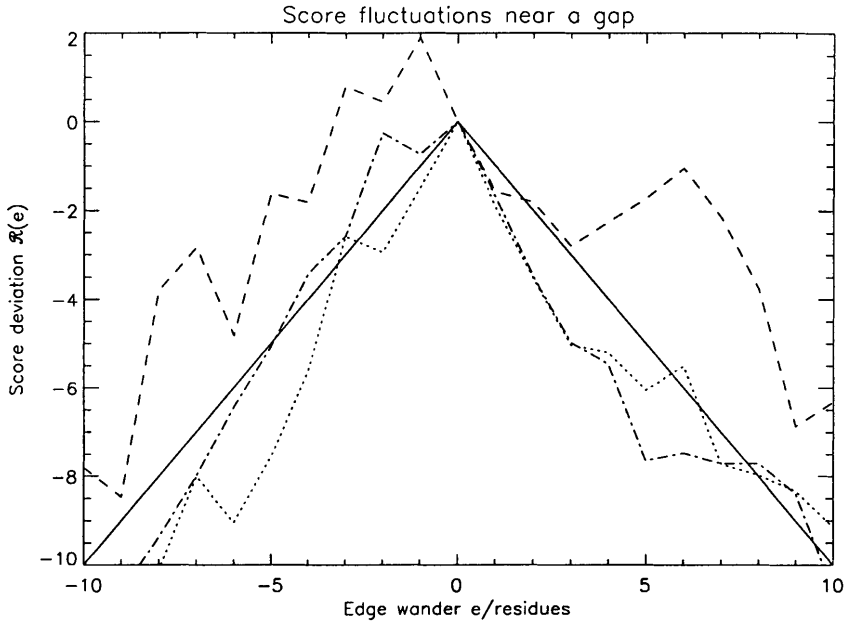

Figure 3.7: Variations in the alignment score when a gap is moved away from its correct position by sliding it along a diagonal. The solid line shows the mean behaviour: on average, the score will decrease as the gap is moved away from its correct position, so the score is maximal at $e = 0$. The dotted lines show examples of the behaviour in specific cases. Due to random fluctuations, the peak of $\mathcal{R}(e)$ may be somewhere away from $e = 0$. This means the optimal-scoring position for the alignment algorithm to place the gap will not be the correct position.

The joint probability distribution function (p.d.f.) $\varsigma(s,t)$ of s and t depends on the joint probability distribution q_{XY} of correlated residue pairs and the prior probability q_X of individual residues, defined in (3.3) and (3.4):

$$\varsigma(s,t) = \sum_{X,Y,Z} q_{XY} q_Z \delta(s - \log \frac{q_{XY}}{q_X q_Y}) \delta(t - \log \frac{q_{XZ}}{q_X q_Z})$$

where $\delta(x)$ is the Kronecker delta function:

$$\delta(x) \equiv \begin{cases} 1 & \text{if } x = 0 \\ 0 & \text{if } x \neq 0 \end{cases}$$

For convenience the scores are here written as log odds-ratios with respect to a "null" model whereby all residues are uncorrelated; this does not affect the final result.

The p.d.f. $\rho(r)$ of $r \equiv s - t$ is derived from $\varsigma(s,t)$:

$$\rho(r) = \sum_t \varsigma(r+t,t) = \sum_{X,Y,Z} q_{XY} q_Z \delta(r - \log \frac{q_{XY} q_Z}{q_{XZ} q_Y}) \tag{3.10}$$

Now consider the Viterbi alignment $\mathbf{a}_{\max}$. Since this is the highest scoring alignment, the *Viterbi edge wander* $e_{\max}$ is given by:

$$e_{\max} = \operatorname*{argmax}_e S(e) = \operatorname*{argmax}_e \mathcal{R}(e)$$

i.e. the edge wander is determined by the behaviour of $\mathcal{R}(e)$. If the peak of $\mathcal{R}(e)$ is ambiguous, so that there are two or more possible values for $\operatorname{argmax}_e \mathcal{R}(e)$, then $e_{\max}$ is defined to be the largest of those values.

Let $\mathcal{E}(e)$ be the p.d.f. of $e_{\max}$:

$$\mathcal{E}(e) = \Pr[e_{\max} = e]$$

Utilising the Markov property of $\mathcal{R}(e)$, factorise $\mathcal{E}(e)$ by splitting the process (3.9) into three parts, cutting at $k = 0$ and $k = e$ and summing over allowable values of the difference $y = \mathcal{R}(e) - \mathcal{R}(0)$:

$$\mathcal{E}(e) =$$
$$\begin{cases} \sum_y \mathcal{C}_L(e+E,0).\mathcal{X}_R(-e,y,0).\mathcal{C}_R(E,-y) & \text{if } e \leq 0 \\ \sum_y \mathcal{C}_L(E,-y).\mathcal{X}_L(e,y,0).\mathcal{C}_R(E-e,0) & \text{if } e > 0 \end{cases}$$

where $\mathcal{C}_L$, $\mathcal{C}_R$, $\mathcal{X}_L$ and $\mathcal{X}_R$ are bounding probabilities defined on sums of r_k ($\mathcal{C}$ signifies a cumulative distribution and $\mathcal{X}$ an exact distribution, and the L and R suffices mean "left of the peak" and "right of the peak"):

$$\mathcal{C}_L(x,z) = \Pr[\forall n \in \{1,2,...,x\} : \sum_{k=1}^{n} r_k \leq z]$$

$$\mathcal{C}_R(x,z) = \Pr[\forall n \in \{1,2,...,x\} : \sum_{k=1}^{n} r_k < z]$$

$$\begin{aligned} \mathcal{X}_L(x,y,z) &= \Pr[\sum_{k=1}^{x}(-r_k) = y \quad \text{and} \\ &\qquad \forall n \in \{1,2,...,x\} : \sum_{k=1}^{n}(-r_k) \leq z] \end{aligned}$$

$$\begin{aligned} \mathcal{X}_R(x,y,z) &= \Pr[\sum_{k=1}^{x}(-r_k) = y \quad \text{and} \\ &\qquad \forall n \in \{1,2,...,x\} : \sum_{k=1}^{n}(-r_k) < z] \end{aligned}$$

The $\mathcal{C}_L$, $\mathcal{C}_R$, $\mathcal{X}_L$ and $\mathcal{X}_R$ can be found by recursive decomposition, separating the first step from the $(x-1)$ succeeding ones:

$$\mathcal{C}_L(x,z) = \begin{cases} \sum_{r \leq z} \rho(r)\mathcal{C}_L(x-1,z-r) & \text{for } x > 0 \\ 1 & \text{for } x = 0 \end{cases} \tag{3.11}$$

$$\mathcal{C}_R(x,z) = \begin{cases} \sum_{r < z} \rho(r)\mathcal{C}_R(x-1,z-r) & \text{for } x > 0 \\ 1 & \text{for } x = 0 \end{cases} \tag{3.12}$$

Score matrix	$\langle \lvert e \rvert \rangle$	p_G	$F_{g=12}$
PAM40	0.326	$2^{-g/2}$	0.99
PAM80	0.688	$2^{-g/2}$	0.98
PAM120	1.246	$2^{-g/2}$	0.96
PAM160	1.884	$2^{-g/2}$	0.94
PAM200	2.573	$2^{-g/3}$	0.68
PAM250	3.888	$2^{-g/3}$	0.53
BLOSUM100	0.794	$2^{-g/3}$	0.90
BLOSUM75	1.332	$2^{-g/2}$	0.96
BLOSUM62	1.826	$2^{-g/2}$	0.94
BLOSUM50	3.286	$2^{-g/3}$	0.60
BLOSUM45	3.671	$2^{-g/3}$	0.56
BLOSUM30	~24	$2^{-g/5}$	-

Table 3.1: Edge wander for various common amino acid substitution matrices.

$$\mathcal{X}_L(x,y,z) = \begin{cases} \sum_{r \geq -z} \rho(r) \mathcal{X}_L(x-1, y+r, z+r) & \text{for } x > 0 \\ \delta(y) & \text{for } x = 0 \end{cases} \tag{3.13}$$

$$\mathcal{X}_R(x,y,z) = \begin{cases} \sum_{r > -z} \rho(r) \mathcal{X}_R(x-1, y+r, z+r) & \text{for } x > 0 \\ \delta(y) & \text{for } x = 0 \end{cases} \tag{3.14}$$

where $\delta(y)$ is the Kronecker delta again.

A program **edge** [1] has been written to calculate the mean absolute edge wander $\langle \lvert e \rvert \rangle$ for various common substitution matrices; the results are listed in Table 3.1. To find the expected fidelity given the mean edge wander, use the following formula:

$$F = 1 - \langle \lvert e \rvert \rangle \left(1 - (1 - p_G)^2\right) \tag{3.15}$$

[1] C++ source code for the **edge** program is available at `http://www.sanger.ac.uk/Users/ihh/edge.html`

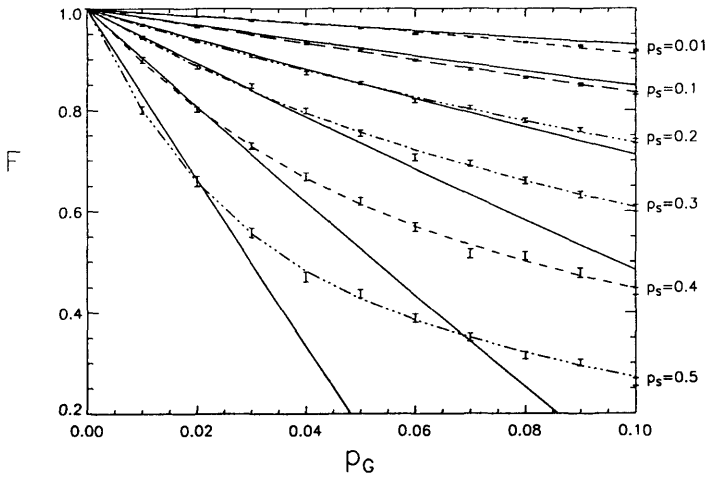

Figure 3.8: The fidelity data of Figure 3.5a (dashed lines), plotted along with the predictions of the edge wander theory (solid lines). Near $p_G \sim 0$, the edge wander theory always slightly overestimates the fidelity. When p_S is small, this trend continues for higher p_G, but for higher p_S (notably $p_S = 0.5$) the edge wander quickly exceeds the mean path fragment length and the theory consequently underestimates the fidelity.

taking p_G to be the observed gap frequency per strand. Alternatively, p_G can be calculated from the gap opening penalty ($-g$, where $g > 0$) using the formulae in the third column of Table 3.1. The values in the final column (labelled $F_{g=12}$) are the expected fidelities when $g = 12$. Note that the prediction for F is independent of the particular gap model being used (e.g. linear or affine). Equations (3.11)-(3.14) describe a random walk with an absorbing barrier and a reflection at the origin. These equations appear amenable to further manipulation to speed up calculations; for example, the distribution (3.10) might be successfully approximated by a more tractable distribution such as a Gaussian.

Figure 3.8 compares the predictions of this section with some of the results from simulation 2. There is a good correspondence between the edge wander predictions and the simulation data.

3.3.7 Estimating the fidelity of a particular alignment

Given a probabilistic model such as the one shown in Figure 3.1, the posterior probability of a particular coupling $(i \diamond j)$ can be calculated:

$$\Pr[i \diamond j] = \sum_{\mathbf{a}:(i \diamond j) \in \mathbf{a}} \Pr[\mathbf{a}]$$

The sum is over all paths that contain this coupling and is straightforward to compute using the Forward-Backward algorithm described in Chapter 2.

Using this result one can write down an expression for the expected overlap $\hat{A}(\mathbf{a})$ between a given alignment $\mathbf{a}$ and paths sampled from the posterior distribution. This is equivalently the expected number of correct matches in $\mathbf{a}$, which is a natural measure of the overall accuracy of $\mathbf{a}$.

$$\hat{A}(\mathbf{a}) = \sum_{(i \diamond j) \in \mathbf{a}} \Pr[i \diamond j]$$

where the sum is over all aligned pairs in $\mathbf{a}$.

It is also possible to write down $\hat{M}$, the expected number of matches in a path sampled from the posterior distribution (and the expected total number of matches in the real alignment):

$$\hat{M} = \sum_{\text{all } (i \diamond j)} \Pr[i \diamond j]$$

The above two quantities are posterior expectations of the numerator and denominator of (3.1). An estimate for the fidelity $\hat{F}(\mathbf{a})$ of a given alignment $\mathbf{a}$ is:

$$\hat{F}(\mathbf{a}) = \frac{\hat{A}(\mathbf{a})}{\hat{M}} \tag{3.16}$$

3.3.8 An optimal accuracy alignment algorithm

Given this new type of score for an alignment, it is possible to find the alignment that maximises this score, and hence has the highest predicted accuracy (by this

defintion of accuracy, of course). The algorithm to do this has been described elsewhere [DEKM98] and is revisited here. The method required is identical to standard dynamic programming, but uses score values given by the posterior probabilities of pair matches; gap costs are not used. The dynamic programming recursion equations are:

$$A(i,j) = \max \begin{cases} A(i-1,j-1) + \Pr[i \diamond j] \\ A(i-1,j) \\ A(i,j-1) \end{cases}$$

and the standard traceback procedure will produce the best alignment [DEKM98]. The structure of this recursion ensures that the returned alignment will be legitimate, and the calculation of the cost function ensures that the alignment is optimised for the sum of the $\Pr[i \diamond j]$ terms along its path. Interestingly the same algorithm works for any sort of gap score; what will change with different scores are the $\Pr[i \diamond j]$ terms themselves, which are obtained from the standard, scoring scheme-specific dynamic programming algorithms referred to above.

An implementation of the optimal accuracy algorithm is available from `http://www.sanger.ac.uk/Users/ihh/optacc.html`

3.3.9 Simulation 3: Evaluation of the optimal accuracy algorithm

In order to test the prediction that the optimal accuracy alignment algorithm outperforms the Viterbi algorithm when the assumed model is correct, the sequence generation and re-alignment procedure of simulation 2 was repeated using the optimal accuracy algorithm.

Figure 3.9a shows the results of these simulations compared with the corresponding data for the Viterbi algorithm from simulation 2. It is clear the optimal accuracy algorithm has a significant advantage. Figure 3.9b is a plot of the expected fidelity (3.16) of these alignments against the measured fidelity. The correspondence is evident, supporting the validity of this particular statistic.

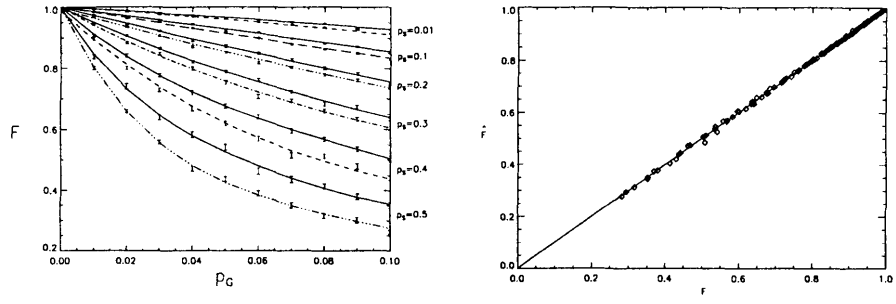

Figure 3.9: Evaluation of the optimal accuracy alignment algorithm. (a) Fidelity data for the Viterbi algorithm (dashed lines) plotted with data for the optimal accuracy algorithm (solid lines). (b) The expected fidelity plotted against the measured fidelity for the data points in (a). The Viterbi data are from simulation 2 (see Figure 3.5a) and the optimal accuracy data from simulation 3.

3.4 Discussion

It has been demonstrated that using a *maximum likelihood* scoring with the dynamic programming algorithm also appears to give maximally faithful alignments. With the aid of alignment fidelity measurements collected using a simulated model of evolution, the dependence of the alignment fidelity on the underlying mutation parameters has been discussed, and an analytic approximation (the *edge wander* approximation) describing this dependence has been presented along with a method for calculating the expected fidelity of a given alignment and an algorithm for finding the expected optimal-accuracy alignment.

These results demonstrate that the edge wander theory is a useful first-order approximation up to large values of p_S. Application of the theory to common substitution matrices predicts the extent of the unrecoverable loss of alignment information. The more distant the similarity, the less accurate we can expect the alignment to be. When aligning sequences diverged by 250 PAMs, for example, one must assume an average error of around 3.9 residues in the positioning of every gap, whereas that expected error is only 1.2 residues at 120 PAMs. In particular, we must not expect alignments for matches in the twilight zone of

detectability to be accurate.

There is a statistical physics analogy that may help to give insight into the edge wander approximation. Consider the variable r whose probability density function $\rho(r)$ is given by (3.10). The mean value of r, $\bar{r} = \langle r \rangle_\rho$, is a relative entropy or Kullback-Leibler divergence between two probability distributions, representing the adjacent diagonals that the Viterbi path could lie on. The variance of r, $\langle (r - \bar{r})^2 \rangle_\rho$, is related to the fluctuations in this entropy-like quantity. The relative sizes of $\langle (r - \bar{r})^2 \rangle_\rho$ and $\bar{r}^2$ indicate the extent of the score fluctuations and equations (3.11)-(3.14) relate this to the error in the gap positioning, i.e. the edge wander. The edge wander approximation essentially assumes that the entropy (score) fluctuations are small and that the Viterbi path is "bound" to the correct path. This approximation is similar to perturbative approaches in statistical physics [LL80]. When edge wander breaks down, a full treatment of the critical scaling phenomena of the path behaviour is required. Terence Hwa, Michael Lässig and Dirk Drasdo [HL96, Hwa96, DHL97b, DHL97a] have published analyses of this problem that apply the theory of the renormalisation group, successfully used in areas of physics as diverse as quantum electrodynamics and chaos theory. The behaviour of the optimal path turns out to be analogous to the pinning of magnetic flux lines by randomly scattered defects in superconductors and the statistical behaviour of *directed polymers* in a random potential, both of which are well-studied by physicists. The renormalisation group is mathematically difficult compared to the probability theory used in this chapter, but it apparently has a lot to offer to the theory of sequence alignment algorithms. A notable result is that the renormalisation group theory predicts an optimal scoring scheme [HL96] that contradicts (3.8). This result is deserving of further investigation; a good starting-point would be to repeat Simulation 1 to greater precision.

The optimal accuracy algorithm described here and in [DEKM98] provides a marked improvement on the Viterbi algorithm. It will be interesting to see

if this improvement carries over to real biological alignments. The simulations presented here also verify that the expected fidelity of an alignment is a useful indicator of alignment accuracy.

The observation that perfect alignment recovery is theoretically unattainable reinforces the idea that for some applications, it may be advantageous to consider a set or *envelope* of suboptimal alignment paths rather than singling out the highest-scoring path. Examples of such envelopes might include only residue couplings whose likelihood exceeded some cutoff value, or be defined by a set of path constraints chosen to maximise the sum of the likelihoods of the paths thus contained. An example of the former type has been proposed by Miyazawa [Miy94]; the issue of alignment reliability has also been addressed by Mevissen and Vingron [MV96].

In conclusion, it is noted once again that many of the results presented here are applicable to any dynamic programming based sequence homology algorithm, not just Needleman-Wunsch with linear gap penalties. Once there is a gap, the score changes involved in moving it as in the edge-wander calculation are the same for affine and linear gap penalties, and also for local and global alignments. It is hoped that the quantitative results for the alignment fidelities will be of use both to researchers in molecular evolution and to users of sequence alignment software.